MODULO INTERRUPÇÕES DO dsPIC30F4011

OBJETIVOS

- Ter noções das interrupções no dsPIC30F4011.
- Aprender sobre a Tabela de vetores de interrupção.
- Configurar registradores para trabalhar com as interrupções.
- Fazer programas com as interrupções externas do dsPIC30F4011.

INTRODUÇÃO

Uma interrupção é a ocorrência de uma condição - um evento - que causa uma suspensão temporária de um programa enquanto que a condição é atendida por outro programa.

As interrupções permitem ao sistema responder assincronamente a um evento e lidar com este enquanto outro programa esta sendo executado. Um sistema de interrupção dá a ilusão de se estar fazendo muitas coisas simultaneamente. Geralmente, a CPU não pode executar mais que uma instrução por vez; mas este pode suspender a execução de um programa temporariamente, e executar outro, e depois retornar ao programa original. Este é parecido a uma função em linguagem C.

A diferença entre uma função e uma interrupção é que a interrupção é uma resposta a um "evento" que acontece assincronamente no programa principal. Não se sabe quando o programa principal vai ser interrompido. O programa que lida com a interrupção é chamada de **rotina de serviço de interrupção (Interrup Service Routine - ISR)**.

O ISR é executado em resposta à interrupção externa ou interna e geralmente executa uma operação de entrada ou saída para um dispositivo. Quando acontece uma interrupção, o programa principal temporariamente suspende sua execução e pula para o ISR; o ISR executa a operação e termina com um "retorno da interrupção" e depois o programa principal continua desde onde ficou.

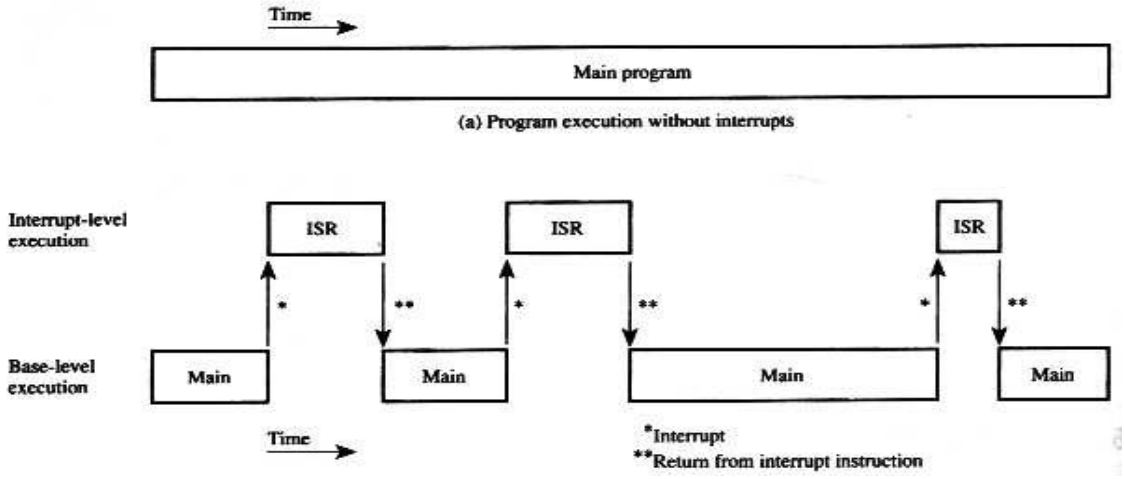

Figura 1. Descrição da execução de um programa sem interrupções (a) e a execução no nível base com interrupções ocasionais, executando as ISRs em nível de interrupção.

Um exemplo típico de interrupção é a entrada manual usando um teclado. Considerando um forno de microondas, onde o programa principal controla o elemento de potência de microondas para cozinhar; ainda cozinhando o sistema deve responder a várias entradas manuais neste caso a porta do forno (quando se abre) ou quando o usuário deixa de pressionar uma tecla, uma interrupção é gerada (o sinal vai de ALTO para BAIXO, talvez) e o programa principal é interrompido. O ISR então lê o código do teclado e muda as condições de cozimento do forno e finaliza passando o controle de novo para o programa principal. O programa principal recomeça onde ficou quando aconteceu a interrupção.

Como se podem ver as entradas manuais acontecem “assincronamente” em intervalos que não podem ser preditas ou controladas pelo software que roda no sistema. **ESTA É UMA INTERRUPÇÃO.**

PINOS DAS INTERRUPÇÕES EXTERNAS DO dsPIC30F4011

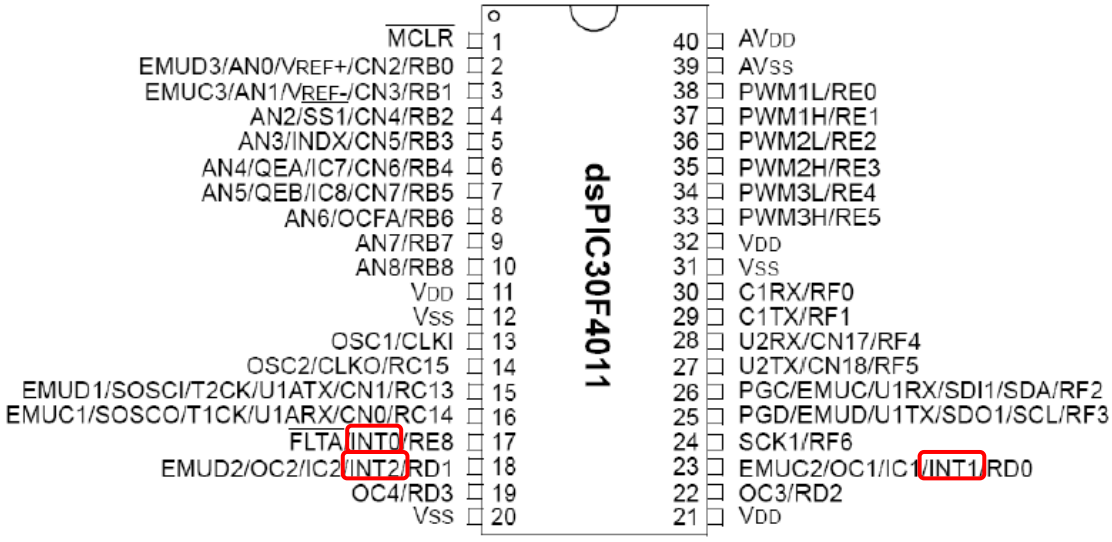

Figura 2. Microcontrolador mostrando os pinos das interrupções externas

ORGANIZAÇÃO DAS INTERRUPÇÕES DO DSPIC30F4011

O dsPIC30F4011 tem 30 fontes de interrupção e 4 exceções de processador (traps), os quais devem ser arbitrados baseados em um esquema de prioridade.

A CPU é responsável de ler a tabela de vetores de interrupção (Interrupt Vector Table – IVT) e transferir o endereço contido no vetor de interrupção para o contador de programa. O vetor de interrupção é transferido desde o barramento de dados de programa para o contador de programa via um multiplexador de 24 bits de largura na entrada do contador de programa. O IVT e a tabela de vetores de interrupção alternada (Alternate Interrupt Vector Table – AIVT) estão localizadas próximos ao começo da memória de programa (0x000004).

INT Number	Vector Number	Interrupt Source
Highest Natural Order Priority		
0	8	INT0 – External Interrupt 0
1	9	IC1 – Input Capture 1
2	10	OC1 – Output Compare 1
3	11	T1 – Timer 1
4	12	IC2 – Input Capture 2
5	13	OC2 – Output Compare 2
6	14	T2 – Timer 2
7	15	T3 – Timer 3
8	16	SPI1
9	17	U1RX – UART1 Receiver
10	18	U1TX – UART1 Transmitter
11	19	ADC – ADC Convert Done
12	20	NVM – NVM Write Complete
13	21	SI2C – I^2C™ Slave Interrupt
14	22	MI2C – I^2C Master Interrupt
15	23	Input Change Interrupt
16	24	INT1 – External Interrupt 1
17	25	IC7 – Input Capture 7
18	26	IC8 – Input Capture 8
19	27	OC3 – Output Compare 3
20	28	OC4 – Output Compare 4
21	29	T4 – Timer4
22	30	T5 – Timer5
23	31	INT2 – External Interrupt 2
24	32	U2RX – UART2 Receiver
25	33	U2TX – UART2 Transmitter
26	34	Reserved
27	35	C1 – Combined IRQ for CAN1
28	36	Reserved
29	37	Reserved
30	38	Reserved
31	39	Reserved
32	40	Reserved
33	41	Reserved
34	42	Reserved
35	43	Reserved
36	44	Reserved
37	45	Reserved
38	46	Reserved
39	47	PWM – PWM Period Match
40	48	QEI – QEI Interrupt
41	49	Reserved
42	50	Reserved
43	51	$\overline{\text{FLTA}}$ – PWM Fault A
44	52	Reserved
45-53	53-61	Reserved
Lowest Natural Order Priority		

O controlador de interrupção é responsável por pré-processar as interrupções e as exceções do processador, apresentando estes para o núcleo do processador. As interrupções periféricas e traps são habilitadas, priorizadas e controladas usando os registradores de funções especiais (SFRs).

IFS0<15:0>, IFS1<15:0>, IFS2<15:0>: Todos os flags de requisição de interrupção estão nestes três registradores. Os flags são setados por seus respectivos periféricos ou sinais externos e são zerados via software.

IFS0 : Interrup Flag Status Register 0					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	INT0IF	External Interrupt 0 Flag Status bit	8	SPI1IF	SPI1 Interrupt Flag Status bit
1	IC1IF	Input Capture Channel 1 Interrupt Flag Status bit	9	U1RXIF	UART1 Receiver Interrupt Flag Status bit
2	OC1IF	Output Compare Channel 1 Interrupt Flag Status bit	10	U1TXIF	UART1 Transmitter Interrupt Flag Status bit
3	T1IF	Timer1 Interrupt Flag Status bit	11	ADIF	A/D Conversion Complete Interrupt Flag Status bit
4	IC2IF	Input Capture Channel 2 Interrupt Flag Status bit	12	NVMIF	Non-Volatile Memory Write Complete Interrupt Flag Status bit
5	OC2IF	Output Compare Channel 2 Interrupt Flag Status bit	13	SI2CIF	I2C Transfer Complete Interrupt Flag Status bit
6	T2IF	Timer2 Interrupt Flag Status bit	14	MI2CIF	I2C Bus Collision Flag Status bit
7	T3IF	Timer3 Interrupt Flag Status bit	15	CNIF	Input Change Notification Flag Status bit

1 = Interrupt request has occurred
0 = Interrupt request has not occurred

IFS1 : Interrup Flag Status Register 1					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	INT1IF	External Interrupt 1 Flag Status bit	8	U2RXIF	UART2 Receiver Interrupt Flag Status bit
1	IC7IF	Input Capture Channel 7 Interrupt Flag Status bit	9	U2TXIF	UART2 Transmitter Interrupt Flag Status bit
2	IC8IF	Input Capture Channel 8 Interrupt Flag Status bit	10	SPI2IF	SPI2 Interrupt Flag Status bit
3	OC3IF	Output Compare Channel 3 Interrupt Flag Status bit	11	C1IF	CAN1 (Combined) Interrupt Flag Status bit
4	OC4IF	Output Compare Channel 4 Interrupt Flag Status bit	12	IC3IF	Input Capture Channel 3 Interrupt Flag Status bit
5	T4IF	Timer4 Interrupt Flag Status bit	13	IC4IF	Input Capture Channel 4 Interrupt Flag Status bit
6	T5IF	Timer5 Interrupt Flag Status bit	14	IC5IF	Input Capture Channel 5 Interrupt Flag Status bit
7	INT2IF	External Interrupt 2 Flag Status bit	15	IC6IF	Input Capture Channel 6 Interrupt Flag Status bit

IFS2 : Interrup Flag Status Register 2					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	OC5IF	Output Compare Channel 5 Interrupt Flag Status bit	8	QEIIF	Quadrature Encoder Interface Interrupt Flag Status bit
1	OC6IF	Output Compare Channel 6 Interrupt Flag Status bit	9	DCIIF	Data Converter Interface Interrupt Flag Status bit
2	OC7IF	Output Compare Channel 7 Interrupt Flag Status bit	10	LVDIF	Programmable Low Voltage Detect Interrupt Flag Status bit
3	OC8IF	Output Compare Channel 8 Interrupt Flag Status bit	11	FLTAIF	Fault A Input Interrupt Flag Status bit
4	INT3IF	External Interrupt 3 Flag Status bit	12	FLTBIF	: Fault B Input Interrupt Flag Status bit
5	INT4IF	External Interrupt 4 Flag Status bit	13		Não Implementado, ler como 0.
6	C2IF	CAN2 (Combined) Interrupt Flag Status bit	14		Não Implementado, ler como 0.
7	PWMIF	Motor Control Pulse Width Modulation Interrupt Flag Status bit	15		Não Implementado, ler como 0.

IEC0<15:0>, IEC1<15:0>, IEC2<15:0>: Todos os bits de controle de habilitação de interrupções são mantidos nestes três registradores. Estes bits de controle são usados para habilitar interrupções individualmente desde sinais periféricos ou externos.

IEC0 : Interrupt Enable Control Register 0					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	INT0IE	External Interrupt 0 Enable bit	8	SPI1IE	SPI1 Interrupt Enable bit
1	IC1IE	Input Capture Channel 1 Interrupt Enable bit	9	U1RXIE	UART1 Receiver Interrupt Enable bit
2	OC1IE	Output Compare Channel 1 Interrupt Enable bit	10	U1TXIE	UART1 Transmitter Interrupt Enable bit
3	T1IE	Timer1 Interrupt Enable bit	11	ADIE	A/D Conversion Complete Interrupt Enable bit
4	IC2IE	Input Capture Channel 2 Interrupt Enable bit	12	NVMIE	Non-Volatile Memory Write Complete Interrupt Enable bit
5	OC2IE	Output Compare Channel 2 Interrupt Enable bit	13	SI2CIE	I2C Transfer Complete Interrupt Enable bit
6	T2IE	Timer2 Interrupt Enable bit	14	MI2CIE	I2C Bus Collision Interrupt Enable bit
7	T3IE	Timer3 Interrupt Enable bit	15	CNIE	Input Change Notification Interrupt Enable bit

1 = Interrupt request enabled
0 = Interrupt request not enabled

IEC1 : Interrupt Enable Control Register 1					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	INT1IE	External Interrupt 1 Enable bit	8	U2RXIE	UART2 Receiver Interrupt Enable bit
1	IC7IE	Input Capture Channel 7 Interrupt Enable bit	9	U2TXIE	UART2 Transmitter Interrupt Enable bit
2	IC8IE	Input Capture Channel 8 Interrupt Enable bit	10	SPI2IE	SPI2 Interrupt Enable bit
3	OC3IE	Output Compare Channel 3 Interrupt Enable bit	11	C1IE	CAN1 (Combined) Interrupt Enable bit
4	OC4IE	Output Compare Channel 4 Interrupt Enable bit	12	IC3IE	Input Capture Channel 3 Interrupt Enable bit
5	T4IE	Timer4 Interrupt Enable bit	13	IC4IE	Input Capture Channel 4 Interrupt Enable bit
6	T5IE	Timer5 Interrupt Enable bit	14	IC5IE	Input Capture Channel 5 Interrupt Enable bit
7	INT2IE	External Interrupt 2 Enable bit	15	IC6IE	Input Capture Channel 6 Interrupt Enable bit

IEC2 : Interrupt Enable Control Register 2					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	OC5IE	Output Compare Channel 5 Interrupt Enable bit	8	QEIIE	Quadrature Encoder Interface Interrupt Enable bit
1	OC6IE	Output Compare Channel 6 Interrupt Enable bit	9	DCIIE	Data Converter Interface Interrupt Enable bit
2	OC7IE	Output Compare Channel 7 Interrupt Enable bit	10	LVDIE	Programmable Low Voltage Detect Interrupt Enable bit
3	OC8IE	Output Compare Channel 8 Interrupt Enable bit	11	FLTAIE	Fault A Interrupt Enable bit
4	INT3IE	External Interrupt 3 Enable bit	12	FLTBIE	Fault B Input Interrupt Enable bit
5	INT4IE	External Interrupt 4 Enable bit	13		Não Implementado, ler como 0.
6	C2IE	CAN2 (Combined) Interrupt Enable bit	14		Não Implementado, ler como 0.
7	PWMIE	Motor Control Pulse Width Modulation Interrupt Enable bit	15		Não Implementado, ler como 0.

IPC0<15:0>... IPC11<7:0> : O nível de prioridade atribuído pelo usuário associado com cada uma das interrupções é mantido nestes doze registradores.

IPC0 : Interrupt Priority Control Register 0					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0-2	**INT0IP<2:0>**	**External Interrupt 0 Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**8-10**	**OC1IP<2:0>**	**Output Compare Channel 1 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
3	-	Não implementado. Ler como 0.	**11**	-	Não implementado. Ler como 0
4-6	**IC1IP<2:0>**	**Input Capture Channel 1 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**12-14**	**T1IP<2:0>**	**Timer1 Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
7	-	Não implementado. Ler como 0.	**15**	-	Não implementado. Ler como 0.

SE ALGUMA INTERRUPÇÃO TEM NÍVEL DE PRIORIDADE IGUAL A 0, É O MESMO DIZER QUE NÃO FOI HABILITADA A INTERRUPÇÃO EM IECx.

IPC1 : Interrupt Priority Control Register 1					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0-2	**IC2IP<2:0>**	**Input Capture Channel 2 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**8-10**	**T2IP<2:0>**	**Timer2 Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
3	-	Não implementado. Ler como 0.	**11**	-	Não implementado. Ler como 0
4-6	**OC2IP<2:0>**	**Output Compare Channel 2 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**12-14**	**T3IP<2:0>**	**Timer3 Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
7	-	Não implementado. Ler como 0.	**15**	-	Não implementado. Ler como 0.

IPC2 : Interrupt Priority Control Register 2					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0-2	**SPI1IP<2:0>**	**SPI1 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**8-10**	**U1TXIP<2:0>**	**UART1 Transmitter Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
3	-	Não implementado. Ler como 0.	**11**	-	Não implementado. Ler como 0
4-6	**U1RXIP<2:0>**	**UART1 Receiver Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**12-14**	**ADIP<2:0>**	**A/D Conversion Complete Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
7	-	Não implementado. Ler como 0.	**15**	-	Não implementado. Ler como 0.

IPC3 : Interrupt Priority Control Register 3					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0-2	**NVMIP<2:0>**	**Non-Volatile Memory Write Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**8-10**	**MI2CIP<2:0>**	**I2C Bus Collision Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
3	-	Não implementado. Ler como 0.	**11**	-	Não implementado. Ler como 0
4-6	**SI2CIP<2:0>**	**I2C Ttransfer Complete Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**12-14**	**CNIP<2:0>**	**Input Change Notification Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
7	-	Não implementado. Ler como 0.	**15**	-	Não implementado. Ler como 0.

IPC4 : Interrupt Priority Control Register 4					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0-2	**INT1IP<2:0>**	**External Interrupt 1 Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**8-10**	**IC8IP<2:0>**	**Input Capture Channel 8 Interrupt Priority bits.** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
3	-	Não implementado. Ler como 0.	**11**	-	Não implementado. Ler como 0
4-6	**IC7IP<2:0>**	**Input Capture Channel 7 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled	**12-14**	**OC3IP<2:0>**	**Output Compare Channel 3 Interrupt Priority bits** 111 = Interrupt is priority 7 (highest priority interrupt) • • • 001 = Interrupt is priority 1 000 = Interrupt source is disabled
7	-	Não implementado. Ler como 0.	**15**	-	Não implementado. Ler como 0.

E ASSIM SUCESSIVAMENTE ATÉ O IPC11. PARA MAIS DETALHES VER O MANUAL DE REFERÊNCIA DO dsPIC30F

INTCON1<15:0>, INTCON2<15:0> : Funcionam como controles de interrupção global. O **INTCON1** contém os flags de controle e status para as exceções do processador. O registrador **INTCON2** controla o comportamento do sinal de requerimento de interrupção externa e o uso do **AIVT**.

INTCON1 : Interrupt Control Register 1					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	-	**Não implementado. Ler como 0.**			
1	OSCFAIL	**Oscillator Failure Trap Status bit** 1 = Oscillator failure trap has occurred 0 = Oscillator failure trap has not occurred	8	COVTE	**Catastrophic Overflow Trap Enable bit** 1 = Trap on catastrophic overflow of Accumulator A or B enabled 0 = Trap disabled
2	STKERR	**Stack Error Trap Status bit** 1 = Stack error trap has occurred 0 = Stack error trap has not occurred	9	OVBTE	**Accumulator B Overflow Trap Enable bit** 1 = Trap overflow of Accumulator B 0 = Trap disabled
3	ADDRERR	**Address Error Trap Status bit** 1 = Address error trap has occurred 0 = Address error trap has not occurred	10	OVATE	**Accumulator A Overflow Trap Enable bit** 1 = Trap overflow of Accumulator A 0 = Trap disabled
4	MATHERR	**Arithmetic Error Status bit** 1 = Overflow trap has occurred 0 = Overflow trap has not occurred	11-14	-	**Não implementado. Ler como 0**
5-7	-	**Não implementado. Ler como 0.**	15	NSTDIS	**Interrupt Nesting Disable bit** 1 = Interrupt nesting is disabled 0 = Interrupt nesting is enabled

Se o bit NSTDIS (INTCON1 <15>) é setado, o aninhamento de interrupção é prevista. Assim se uma interrupção está sendo executada, pode ser processada uma nova interrupção, sempre que esta nova interrupção tenha uma maior prioridade.

INTCON2 : Interrupt Control Register 2					
Bit	**Nome**	**Descrição**	**Bit**	**Nome**	**Descrição**
0	INT0EP	**External Interrupt #0 Edge Detect Polarity Select bit** 1 = Interrupt on negative edge 0 = Interrupt on positive edge	4	INT4EP	**External Interrupt #4 Edge Detect Polarity Select bit** 1 = Interrupt on negative edge 0 = Interrupt on positive edge
1	INT1EP	**External Interrupt #1 Edge Detect Polarity Select bit** 1 = Interrupt on negative edge 0 = Interrupt on positive edge	5-13	-	**Não implementado. Ler como 0.**
2	INT2EP	**External Interrupt #2 Edge Detect Polarity Select bit** 1 = Interrupt on negative edge 0 = Interrupt on positive edge	14	DISI	**DISI Instruction Status bit** 1 = DISI instruction is active 0 = DISI is not active
3	INT3EP	**External Interrupt #3 Edge Detect Polarity Select bit** 1 = Interrupt on negative edge 0 = Interrupt on positive edge	15	ALTIVT	**Enable Alternate Interrupt Vector Table bit** 1 = Use alternate vector table 0 = Use standard (default) vector table

A INSTRUÇÃO DISI (SETADA PARA 1) PODE SER USADA PARA DESABILITAR O PROCESSAMENTO DAS INTERRUPÇÕES DESDE A PRIORIDADE 6 PARA ABAIXO.

TABLE 5-2: INTERRUPT CONTROLLER REGISTER MAP(1)

SFR Name	ADR	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Reset State
INTCON1	0080	NSTDIS	—	—	—	—	OVATE	OVBTE	COVTE	—	—	—	MATHERR	ADDRERR	STKERR	OSCFAIL	—	0000 0000 0000 0000
INTCON2	0082	ALTIVT	DISI	—	—	—	—	—	—	—	—	—	—	—	INT2EP	INT1EP	INT0EP	0000 0000 0000 0000
IFS0	0084	CNIF	MI2CIF	SI2CIF	NVMIF	ADIF	U1TXIF	U1RXIF	SPI1IF	T3IF	T2IF	OC2IF	IC2IF	T1IF	OC1IF	IC1IF	INT0IF	0000 0000 0000 0000
IFS1	0086	—	—	—	—	C1IF	—	U2TXIF	U2RXIF	INT2IF	T5IF	T4IF	OC4IF	OC3IF	IC8IF	IC7IF	INT1IF	0000 0000 0000 0000
IFS2	0088	—	—	—	—	FLTAIF	—	—	QEIIF	PWMIF	—	—	—	—	—	—	—	0000 0000 0000 0000
IEC0	008C	CNIE	MI2CIE	SI2CIE	NVMIE	ADIE	U1TXIE	U1RXIE	SPI1IE	T3IE	T2IE	OC2IE	IC2IE	T1IE	OC1IE	IC1IE	INT0IE	0000 0000 0000 0000
IEC1	008E	—	—	—	—	C1IE	—	U2TXIE	U2RXIE	INT2IE	T5IE	T4IE	OC4IE	OC3IE	IC8IE	IC7IE	INT1IE	0000 0000 0000 0000
IEC2	0090	—	—	—	—	FLTAIE	—	—	QEIIE	PWMIE	—	—	—	—	—	—	—	0000 0000 0000 0000
IPC0	0094	—	T1IP<2:0>			—	OC1IP<2:0>			—	IC1IP<2:0>			—	INT0IP<2:0>			0100 0100 0100 0100
IPC1	0096	—	T3IP<2:0>			—	T2IP<2:0>			—	OC2IP<2:0>			—	IC2IP<2:0>			0100 0100 0100 0100
IPC2	0098	—	ADIP<2:0>			—	U1TXIP<2:0>			—	U1RXIP<2:0>			—	SPI1IP<2:0>			0100 0100 0100 0100
IPC3	009A	—	CNIP<2:0>			—	MI2CIP<2:0>			—	SI2CIP<2:0>			—	NVMIP<2:0>			0100 0100 0100 0100
IPC4	009C	—	OC3IP<2:0>			—	IC8IP<2:0>			—	IC7IP<2:0>			—	INT1IP<2:0>			0100 0100 0100 0100
IPC5	009E	—	INT2IP<2:0>			—	T5IP<2:0>			—	T4IP<2:0>			—	OC4IP<2:0>			0100 0100 0100 0100
IPC6	00A0	—	C1IP<2:0>			—	—	—	—	—	U2TXIP<2:0>			—	U2RXIP<2:0>			0100 0000 0100 0100
IPC7	00A2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000 0000 0000 0000
IPC8	00A4	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000 0000 0000 0000
IPC9	00A6	—	PWMIP<2:0>			—	—	—	—	—	—	—	—	—	—	—	—	0100 0000 0100 0100
IPC10	00A8	—	FLTAIP<2:0>			—	—	—	—	—	—	—	—	—	QEIIP<2:0>			0100 0000 0000 0100
IPC11	00AA	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	0000 0000 0000 0000

Legend: — = unimplemented bit, read as '0'

Note 1: Refer to the "*dsPIC30F Family Reference Manual*" (DS70046) for descriptions of register bit fields.

ESCREVENDO UM INTERRUPT SERVICE ROUTINE - ISR

As guias para escrever os ISR´s são:

Declarar os ISR´s sem parâmetros e um tipo de retorno void (obrigatório). Um ISR é parecido a alguma outra função em C em que este pode ter variáveis locais e variáveis de acesso global. Portanto, um ISR necessita ser declarada sem parâmetros e não retorna valores. Isto é necessário porque o ISR, em resposta a uma interrupção de hardware ou trap, é invocada assincronamente no programa principal (isto é, este não é chamado de forma normal, então parâmetros de retorno de valores não são aplicáveis).

Os ISR´s poderiam ser invocados através de uma interrupção de hardware ou trap e não desde uma função (obrigatório). Um ISR usa uma instrução de "retorno desde a interrupção" (return from interrupt – RETFIE) para sair da função ao invés que a instrução normal RETURN. Usando uma instrução RETFIE fora do contexto pode corromper recursos do processador, tal como o registrador de status.

Finalmente, ISR´s poderiam não chamar outras funções (recomendado). Esta recomendação é por causa da latência (tempo entre a chamada da interrupção e a primeira instrução a ser executada dentro do ISR).

A sintaxe para escrever ISR´s é:

```
void interrupt() iv IVT_ADDR_U1RXINTERRUPT ics ICS_AUTO {
  // Código da rotina de serviço de interrupção.
}
```

Onde:

- **iv** – palavra reservada que informa ao compilador que esta é uma rotina de serviço de interrupção..
- **IVT_ADDR_U1RXINTERRUPT** – Vetor de interrupção apropriado.

- **ics** Interrupt Context Saving; Salvando o context da interrupção (Interrupt Context Saving) pode ser executada em várias formas:
    1. `ICS_OFF` – Não salva o context.
    2. `ICS_AUTO` – O compilador escolhe se salvar o contexto será executado ou não.

Exemplo de uma rotina de interrupção.

```
//  rotina de interrupção
void Timer1Int() iv IVT_ADDR_T1INTERRUPT ics ICS_AUTO {
  //** Isto é necessário para limpar manualmente o flag de
interrupção.
  IFS0 = IFS0 & 0xFFF7;   // Limpa TMR1IF

  //** O código do usuário começa aqui
  LATB = ~ PORTB;          // Inverte PORTB
  //** O código do usuário finaliza aqui
}
```

ESCREVENDO O VETOR DE INTERRUPÇÃO

INTERRUPT VECTORS

IRQ#	Vector Function	Primary Name	Alternate Name
n/a	Reserved	_ReservedTrap0	_AltReservedTrap0
n/a	Oscillator fail trap	_OscillatorFail	_AltOscillatorFail
n/a	Address error trap	_AddressError	_AltAddressError
n/a	Stack error trap	_StackError	_AltStackError
n/a	Math error trap	_MathError	_AltMathError
n/a	Reserved	_ReservedTrap5	_AltReservedTrap5
n/a	Reserved	_ReservedTrap6	_AltReservedTrap6
n/a	Reserved	_ReservedTrap7	_AltReservedTrap7
0	INT0-External interrupt 0	_INT0Interrupt	_AltINT0Interrupt
1	IC1-Input capture 1	_IC1Interrupt	_AltIC1Interrupt
2	OC1-Output compare 1	_OC1Interrupt	_AltOC1Interrupt
3	TMR1-Timer 1	_T1Interrupt	_AltT1Interrupt
4	IC2-Input capture 2	_IC2Interrupt	_AltIC2Interrupt
5	OC2-Output compare 2	_OC2Interrupt	_AltOC2Interrupt
6	TMR2-Timer 2	_T2Interrupt	_AltT2Interrupt
7	TMR3-Timer 3	_T3Interrupt	_AltT3Interrupt
8	SPI1-Serial peripheral interface 1	_SPI1Interrupt	_AltSPI1Interrupt
9	UART1RX-UART1 Receiver	_U1RXInterrupt	_AltU1RXInterrupt
10	UART1TX-UART1 Transmitter	_U1TXInterrupt	_AltU1TXInterrupt
11	ADC-ADC convert done	_ADCInterrupt	_AltADCInterrupt
12	NVM-NVM write complete	_NVMInterrupt	_AltNVMInterrupt
13	Slave I^2C Interrupt	_SI2CInterrupt	_AltSI2CInterrupt
14	Master I^2C Interrupt	_MI2CInterrupt	_AltMI2CInterrupt
15	CN-Input change interrupt	_CNInterrupt	_AltCNInterrupt
16	INT1-External interrupt 1	_INT1Interrupt	_AltINT1Interrupt
17	IC7-Input capture 7	_IC7Interrupt	_AltIC7Interrupt
18	IC8-Input capture 8	_IC8Interrupt	_AltIC8Interrupt
19	OC3-Output compare 3	_OC3Interrupt	_AltOC3Interrupt
20	OC4-Output compare 4	_OC4Interrupt	_AltOC4Interrupt
21	TMR4-Timer 4	_T4Interrupt	_AltT4Interrupt
22	TMR5-Timer 5	_T5Interrupt	_AltT5Interrupt
23	INT2-External interrupt 2	_INT2Interrupt	_AltINT2Interrupt

INTERRUPT VECTORS (CONTINUED)

IRQ#	Vector Function	Primary Name	Alternate Name
24	UART2RX-UART2 receiver	_U2RXInterrupt	_AltU2RXInterrupt
25	UART2TX-UART2 transmitter	_U2TXInterrupt	_AltU2TXInterrupt
26	SPI2-Serial peripheral interface 2	_SPI2Interrupt	_AltSPI2Interrupt
27	CAN1-Combined IRQ	_C1Interrupt	_AltC1Interrupt
28	IC3-Input capture 3	_IC3Interrupt	_AltIC3Interrupt
29	IC4-Input capture 4	_IC4Interrupt	_AltIC4Interrupt
30	IC5-Input capture 5	_IC5Interrupt	_AltIC5Interrupt
31	IC6-Input capture 6	_IC6Interrupt	_AltIC6Interrupt
32	OC5-Output compare 5	_OC5Interrupt	_AltOC5Interrupt
33	OC6-Output compare 6	_OC6Interrupt	_AltOC6Interrupt
34	OC7-Output compare 7	_OC7Interrupt	_AltOC7Interrupt
35	OC8-Output compare 8	_OC8Interrupt	_AltOC8Interrupt
36	INT3-External interrupt 3	_INT3Interrupt	_AltINT3Interrupt
37	INT4-External interrupt 4	_INT4Interrupt	_AltINT4Interrupt
38	CAN2-Combined IRQ	_C2Interrupt	_AltC2Interrupt
39	PWM-PWM period match	_PWMInterrupt	_AltPWMInterrupt
40	QEI-Position counter compare	_QEIInterrupt	_AltQEIInterrupt
41	DCI-CODEC transfer done	_DCIInterrupt	_AltDCIInterrupt
42	PLVD-Low voltage detect	_LVDInterrupt	_AltLVDInterrupt
43	FLTA-MPWM fault A	_FLTAInterrupt	_AltFLTAInterrupt
44	FLTB-MPWM fault B	_FLTBInterrupt	_AltFLTBInterrupt
45	Reserved	_Interrupt45	_AltInterrupt45
46	Reserved	_Interrupt46	_AltInterrupt46
47	Reserved	_Interrupt47	_AltInterrupt47
48	Reserved	_Interrupt48	_AltInterrupt48
49	Reserved	_Interrupt49	_AltInterrupt49
50	Reserved	_Interrupt50	_AltInterrupt50
51	Reserved	_Interrupt51	_AltInterrupt51
52	Reserved	_Interrupt52	_AltInterrupt52
53	Reserved	_Interrupt53	_AltInterrupt53

EXEMPLOS

1. Qual é o código para habilitar a interrupção externa 1 e o timer2 do dsPIC30F?

IFS0 = 0; //aqui se encontra o bit de flag de status da interrupção do Timer 2.
IFS1 = 0; //aqui se encontra o bit de flag de status da interrupção Externa 1.
IEC0bits.T2IE = 1; //Habilita a interrupção do timer 2.
IEC1bits.INT1IE = 1; //Habilita a interrupção externa 1.

2. Qual é o código para habilitar a interrupção externa 1 e ativar esta na borda de descida? E para a interrupção externa 2 que se ativa na borda de subida? Responder tudo em um fragmento de código.

IFS1 = 0; //aqui se encontra o bit de flag de status da interrupção Externa 1 e 2.
IEC1bits.INT1IE = 1; //Habilita a interrupção externa 1.
IEC1bits.INT2IE = 1; //Habilita a interrupção externa 2.
INTCON2bits.INT1EP = 1; //Se ativa na borda negativa (descida)
INTCON2bits.INT2EP = 0; //Se ativa na borda positiva (subida)

3. Utilizando a tabela de vetores de interrupção, escrever um programa que inclua o ISR (só o cabeçalho) para uma interrupção externa 0 que se ativa na borda de subida.

void INT0Int() iv IVT_ADDR_INT0INTERRUPT
{.
....
}
void main (void)
{
IFS0 = 0; //aqui se encontra o bit de flag de status da interrupção Externa 0
IEC0bits.INT0IE = 1; //Habilita a interrupção externa 0
INTCON2bits.INT0EP = 0; //Se ativa na borda positiva (subida)
}

4. Toda vez que pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0, na porta B onde estão ligados os leds deve aparecer a contagem do número de vezes que foi pressionada a tecla. Fazer o programa que execute este enunciado usando as interrupções.

*/

```
//************* ISR da interrupção externa 0 *
void INT0Int() iv IVT_ADDR_INT0INTERRUPT
{
  LATB++;
  IFS0bits.INT0IF = 0;
}
//************* Programa Principal ***********
void main ()
{
  ADPCFG = 0xFFFF;
```

```
  TRISB=0; //a PORTB como saída
  TRISE = 0x0100; //entrada INT0=RE8

  IFS0 = 0;      //desabilita o flag desta interrupção
  IEC0bits.INT0IE = 1; //Habilitamos INT0
  INTCON2bits.INT0EP = 0; //Borda positiva
  while( 1 ) ; // Loop infinito

}
```

5. Toda vez que pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0, a porta B ligará e desligara todos os leds com um intervalo de tempo de 250 ms. Quando se pressionar e soltar novamente o piscar dos leds para e assim sucessivamente. Fazer o programa que execute este enunciado usando as interrupções.

```
unsigned char cont = 0;
//************** ISR da interrupção externa 0 *
void INT0Int() iv IVT_ADDR_INT0INTERRUPT
{
  cont++;
  IFS0bits.INT0IF = 0;
}
//** Programa Principal ******
void main ()
{
ADPCFG = 0xFFFF;
TRISB=0; //a PORTB como saída
TRISE = 0x0100; //entrada INT0=RE8
IFS0 = 0;
IEC0bits.INT0IE = 1;//Habilita INT0
INTCON2bits.INT0EP = 0; //Borda positiva
  while( 1 )
  {
    if (cont == 1)
      {
        LATB = 255;
        Delay_ms(250);
        LATB = 0;
        Delay_ms(250);
      }
    if (cont ==0 || cont > 1)
           {
             LATB = 0;
             cont = 0;
           }
   }

}
```

6. Se quer projetar um sistema com as seguintes características:
   a) Quando pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0 uma primeira vez, os displays (1, 2, 3 e 4) deverão mostrar a palavra “HOLA” piscando a intervalos de tempo de 250 ms.
   b) Quando pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0 uma segunda vez, o item a) para onde estiver e o display 1 começa a contar de 0 a 9 sem parar a intervalos de 500 ms.
   c) Quando pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0 uma terceira vez, o item b) para onde estiver e o display 1 começa a contar de forma decrescente sem parar a

intervalos de 500 ms.

d) Na quarta vez que pressionar e soltar a tecla ligada ao pino onde se encontra a entrada da interrupção externa 0, o item c) para onde estiver e começa o item a) e assim sucessivamente. Levar em consideração que a interrupção externa 0 se ativa na borda de descida. Fazer o programa que execute este enunciado usando as interrupções.

Solução:

```
unsigned char cont = 0;
unsigned int i;
unsigned int num0_9[10]={63,6,91,79,102,109,125,7,127,111};
//0,1,2,3,4,5,6,7,8,9
//unsigned int num9_0[10]={111,127,7,125,109,102,79,91,6,63};
//9,8,7,6,5,4,3,2,1,0

//************* ISR da interrupção externa 0 *
void EXT0Int() iv IVT_ADDR_INT0INTERRUPT
{
  cont++;
  IFS0bits.INT0IF = 0;
}
//** Programa Principal ******
void main ()
{
ADPCFG = 0xFFFF;
//Configurando pinos de saída para os displays
   TRISC.B13 = 0;      //Display 1
   TRISC.B14 = 0;      //Display 2
   TRISD.B2 = 0;       //Display 3
   TRISD.B0 = 0;       //Display 4
//Desligando os displays
   LATC.B13 = 0;
   LATC.B14 = 0;
   LATD.B2 = 0;
   LATD.B0 = 0;

TRISB=0; //a PORTB como saída
TRISE = 0x0100; //entrada INT0=RE8
IFS0 = 0;
IEC0bits.INT0IE = 1;//Habilita INT0
INTCON2bits.INT0EP = 0; //Borda positiva ou subida

  while( 1 )
  {
    switch (cont)
        {
                case 0: LATC.B13 = 0;
                        LATC.B14 = 0;
                        LATD.B2 = 0;
                        LATD.B0 = 0;
                        break;
                case 1: i=50;
                        while(i  !=  0)  //Enquanto  não  passar  500  ms  fica
mostrando
                        //contagem nos displays de 7 segmentos
                        {
                          LATD.B0 = 1; //habilita os displays
                          LATB=0x76; //letra H
                          Delay_ms(2); //atraso de 2 milisegundos
                          LATD.B0 = 0; //desabilita display do milhar

                          LATD.B2 = 1; //habilita os displays
                          LATB=0x3F; //letra O
                          Delay_ms(2); //atraso de 2 milisegundos
                          LATD.B2 = 0; //desabilita display das centenas
```

```
                        LATC.B14 = 1; //habilita os displays
                        LATB=0x38; //letra L
                        Delay_ms(2); //atraso de 2 milisegundos
                        LATC.B14 = 0; //desabilita display das dezenas

                        LATC.B13 = 1; //habilita os displays
                        LATB=0x77; //letra A
                        Delay_ms(2); //atraso de 2 milisegundos
                        LATC.B13 = 0; //desabilita display das unidades

                      i--;
                      if (cont == 2){i=0; break;} //se foi pressionada
novamente a tecla e cont foi para 2
                      }
                      LATC.B13 = 0;
                      LATC.B14 = 0;
                      LATD.B2 = 0;
                      LATD.B0 = 0;
                      if (cont == 2){break;} //se foi pressionada novamente
a tecla e cont foi para 2
                      Delay_ms(250);
                      i = 0;
                      break;
              case 2: LATC.B13 = 0;
                      LATC.B14 = 0;
                      LATD.B2 = 0;
                      LATD.B0 = 0;

                      LATC.B13 = 1;
                      while(1)
                      {
                        LATB = num0_9[i];
                        if (i==9){i= -1;}
                        i++;
                        if (cont == 3){break;} //se foi pressionada
novamente a tecla e cont foi para 3
                        Delay_ms(500);
                        break;
                      }
                       break;
              case 3: LATC.B13 = 0;
                      LATC.B14 = 0;
                      LATD.B2 = 0;
                      LATD.B0 = 0;

                      LATC.B13 = 1;
                       while(1)
                       {
                         LATB = num0_9[i];
                         if (i==0){i= 10;}
                         i--;
                         if (cont == 4){break;} //se foi pressionada
novamente a tecla e cont foi para 4
                         Delay_ms(500);
                         break;
                       }
                       break;
              default: LATC.B13 = 0;
                       LATC.B14 = 0;
                       LATD.B2 = 0;
                       LATD.B0 = 0;
                       cont = 1;
                       break;
              }
      }
}
```